

SL Solar Raster Analysis: Working with Large Rasters

Contributed by Bert Granberg
06, Aug. 2009
Last Updated 10, Aug. 2009

Occasionally in the IT and GIS worlds we are faced with conceptually simple tasks that need to be performed at a scale in which the 'out of the box' toolsets don't execute well. Here is one such problem and its solution.

Problem:

For Salt Lake City's Solar Cities USA project, 1 meter resolution LIDAR data was used in conjunction with ESRI's very cool Solar Analysis toolkit to generate a raster analysis grid for total sunlit hours and total solar radiation for the entire city, for each month of the year. A sunlit grid example for a 3 block area centered on the Salt Lake City/County building is shown above.

In all we're talking about 24 grids, each with approximately 500 million cells. Generating these datasets took weeks and was only possible due to the clever Python work by Kevin Bell that allowed for the data to be built in smaller tiles that considered the neighboring tiles (to account for nearby shadow casting features).

The next problem to solve was how to store this data so that it could be quickly queried by a web service to provide information about solar resources on properties and structures throughout the city. After long investigation, AGRC has concluded that no one is going to wait around for the results of 24 raster grid queries...its just too slow. The old proverb of 'raster is faster' seems to apply much more to map algebra type overlay operations than to 'finding a needle in a haystack' (which is what getting the solar profile for a residential rooftop from these 24 large datasets is all about).

Solution:

The proposed solution is to convert the raster data to a single file geodatabase point-based feature class with 24 attributes describing the monthly sunlit and solar radiation values. The sunlit dataset will give users a good idea of the solar resources for rack mounted PV & water heat systems, landscaping etc. The solar radiation values will give users an idea of the solar energy potential for existing surfaces (flush mount PV, passive cooling strategy, et. The output point dataset containing all data from the 24 raster grids will have to be in a FGDB storage format to allow for large file sizes and not rack up big network storage costs.

The final problem is how to do this. Again, out of the box tools failed due to wait time and memory shortages. The solution seems to be working with the ArcObjects IRasterCursor object and iterating through the cells of all 24 rasters reading in only 128 rows of the 24 rasters into memory at a time. This is precisely what IRasterCursor does (see code below). The product, when finished will only contain points that have data associated with them and should be around 250 million points. It look like it'll take about 24 hours to run and seems to run in fairly linear time relative to the total number of grid cells.

Stay tuned for more on this sometimes bizarre undertaking.

Update 8/10/09: The dataset is done processing and seems to be working albeit a little slowly. To address this I am recalculating the spatial grid size to 30 meters. ArcCatalog suggested 1.4 meters (which would give 9 points per grid cell) but this seems like it would be too many grid cells (both for calculation time for building the index and also for processing a typical residential or commercial area query). I am calculating the spatial index, originally 3000 meters, to 30 meters

```
Public Sub monthlySolarRastersToFGDBPoints()
```

```
    Dim pMxDoc As IMxDocument
    Dim pMap As IMap
```

```
    Dim pFLayer As IFeatureLayer
    Dim pFC As IFeatureClass
    Dim pFCursor As IFeatureCursor
    Dim pFeatureBuffer As IFeatureBuffer
```

```
    Dim pRasterLayer As IRasterLayer
    Dim pRaster As IRaster
```

```
Dim pRasterCursorDur1 As IRasterCursor
Dim pRasterCursorDur2 As IRasterCursor
Dim pRasterCursorDur3 As IRasterCursor
Dim pRasterCursorDur4 As IRasterCursor
Dim pRasterCursorDur5 As IRasterCursor
Dim pRasterCursorDur6 As IRasterCursor
Dim pRasterCursorDur7 As IRasterCursor
Dim pRasterCursorDur8 As IRasterCursor
Dim pRasterCursorDur9 As IRasterCursor
Dim pRasterCursorDur10 As IRasterCursor
Dim pRasterCursorDur11 As IRasterCursor
Dim pRasterCursorDur12 As IRasterCursor
Dim pRasterCursorSol1 As IRasterCursor
Dim pRasterCursorSol2 As IRasterCursor
Dim pRasterCursorSol3 As IRasterCursor
Dim pRasterCursorSol4 As IRasterCursor
Dim pRasterCursorSol5 As IRasterCursor
Dim pRasterCursorSol6 As IRasterCursor
Dim pRasterCursorSol7 As IRasterCursor
Dim pRasterCursorSol8 As IRasterCursor
Dim pRasterCursorSol9 As IRasterCursor
Dim pRasterCursorSol10 As IRasterCursor
Dim pRasterCursorSol11 As IRasterCursor
Dim pRasterCursorSol12 As IRasterCursor
Dim pPixelBlockDur1 As IPixelBlock
Dim pPixelBlockDur2 As IPixelBlock
Dim pPixelBlockDur3 As IPixelBlock
Dim pPixelBlockDur4 As IPixelBlock
Dim pPixelBlockDur5 As IPixelBlock
Dim pPixelBlockDur6 As IPixelBlock
Dim pPixelBlockDur7 As IPixelBlock
Dim pPixelBlockDur8 As IPixelBlock
Dim pPixelBlockDur9 As IPixelBlock
Dim pPixelBlockDur10 As IPixelBlock
Dim pPixelBlockDur11 As IPixelBlock
Dim pPixelBlockDur12 As IPixelBlock
Dim pPixelBlockSol1 As IPixelBlock
Dim pPixelBlockSol2 As IPixelBlock
Dim pPixelBlockSol3 As IPixelBlock
Dim pPixelBlockSol4 As IPixelBlock
Dim pPixelBlockSol5 As IPixelBlock
Dim pPixelBlockSol6 As IPixelBlock
Dim pPixelBlockSol7 As IPixelBlock
Dim pPixelBlockSol8 As IPixelBlock
Dim pPixelBlockSol9 As IPixelBlock
Dim pPixelBlockSol10 As IPixelBlock
Dim pPixelBlockSol11 As IPixelBlock
Dim pPixelBlockSol12 As IPixelBlock

Dim x, y, startX, startY, currX, currY, increment, pbCount, flushCount As Long
Dim pPoint As IPoint

Set pMxDoc = ThisDocument
Set pMap = pMxDoc.FocusMap

Set pFLayer = pMap.Layer(0)
Set pFC = pFLayer.FeatureClass
Set pFCursor = pFC.Insert(True)

Set pRasterLayer = pMap.Layer(1)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur1 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(2)
Set pRaster = pRasterLayer.Raster
```

```
Set pRasterCursorDur2 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(3)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur3 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(4)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur4 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(5)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur5 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(6)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur6 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(7)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur7 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(8)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur8 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(9)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur9 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(10)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur10 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(11)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur11 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(12)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorDur12 = pRaster.CreateCursor
```

```
Set pRasterLayer = pMap.Layer(13)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol1 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(14)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol2 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(15)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol3 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(16)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol4 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(17)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol5 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(18)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol6 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(19)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol7 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(20)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol8 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(21)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol9 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(22)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol10 = pRaster.CreateCursor
Set pRasterLayer = pMap.Layer(23)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol11 = pRaster.CreateCursor
```

```
Set pRasterLayer = pMap.Layer(24)
Set pRaster = pRasterLayer.Raster
Set pRasterCursorSol12 = pRaster.CreateCursor
```

```
startX = 407999.5
startY = 4521978.5
currX = startX
currY = startY
increment = 1
pbCount = 0
flushCount = 0
'need an insertcursor & flush
```

Do

```
If pbCount > 0 Then
  pRasterCursorDur2.Next
  pRasterCursorDur3.Next
  pRasterCursorDur4.Next
  pRasterCursorDur5.Next
  pRasterCursorDur6.Next
  pRasterCursorDur7.Next
  pRasterCursorDur8.Next
  pRasterCursorDur9.Next
  pRasterCursorDur10.Next
  pRasterCursorDur11.Next
  pRasterCursorDur12.Next
  pRasterCursorSol1.Next
  pRasterCursorSol2.Next
  pRasterCursorSol3.Next
  pRasterCursorSol4.Next
  pRasterCursorSol5.Next
  pRasterCursorSol6.Next
  pRasterCursorSol7.Next
  pRasterCursorSol8.Next
  pRasterCursorSol9.Next
  pRasterCursorSol10.Next
  pRasterCursorSol11.Next
  pRasterCursorSol12.Next
End If
```

```
Set pPixelBlockDur1 = pRasterCursorDur1.PixelBlock
Set pPixelBlockDur2 = pRasterCursorDur2.PixelBlock
Set pPixelBlockDur3 = pRasterCursorDur3.PixelBlock
Set pPixelBlockDur4 = pRasterCursorDur4.PixelBlock
Set pPixelBlockDur5 = pRasterCursorDur5.PixelBlock
Set pPixelBlockDur6 = pRasterCursorDur6.PixelBlock
Set pPixelBlockDur7 = pRasterCursorDur7.PixelBlock
Set pPixelBlockDur8 = pRasterCursorDur8.PixelBlock
Set pPixelBlockDur9 = pRasterCursorDur9.PixelBlock
Set pPixelBlockDur10 = pRasterCursorDur10.PixelBlock
Set pPixelBlockDur11 = pRasterCursorDur11.PixelBlock
Set pPixelBlockDur12 = pRasterCursorDur12.PixelBlock
Set pPixelBlockSol1 = pRasterCursorSol1.PixelBlock
Set pPixelBlockSol2 = pRasterCursorSol2.PixelBlock
Set pPixelBlockSol3 = pRasterCursorSol3.PixelBlock
Set pPixelBlockSol4 = pRasterCursorSol4.PixelBlock
Set pPixelBlockSol5 = pRasterCursorSol5.PixelBlock
Set pPixelBlockSol6 = pRasterCursorSol6.PixelBlock
Set pPixelBlockSol7 = pRasterCursorSol7.PixelBlock
Set pPixelBlockSol8 = pRasterCursorSol8.PixelBlock
Set pPixelBlockSol9 = pRasterCursorSol9.PixelBlock
Set pPixelBlockSol10 = pRasterCursorSol10.PixelBlock
Set pPixelBlockSol11 = pRasterCursorSol11.PixelBlock
Set pPixelBlockSol12 = pRasterCursorSol12.PixelBlock
```

```
For y = 0 To pPixelBlockDur1.Height - 1
  'Debug.Print Now
```

```
For x = 0 To pPixelBlockDur1.Width - 1
```

```
  Set pFeatureBuffer = pFC.CreateFeatureBuffer
  Set pPoint = New Point
  pPoint.x = currX
  pPoint.y = currY
  Set pFeatureBuffer.Shape = pPoint
```

```
  If pPixelBlockSol1.GetVal(0, x, y) > 0 Then
```

```
    pFeatureBuffer.Value(2) = pPixelBlockSol1.GetVal(0, x, y)
    pFeatureBuffer.Value(3) = pPixelBlockSol2.GetVal(0, x, y)
    pFeatureBuffer.Value(4) = pPixelBlockSol3.GetVal(0, x, y)
    pFeatureBuffer.Value(5) = pPixelBlockSol4.GetVal(0, x, y)
    pFeatureBuffer.Value(6) = pPixelBlockSol5.GetVal(0, x, y)
    pFeatureBuffer.Value(7) = pPixelBlockSol6.GetVal(0, x, y)
    pFeatureBuffer.Value(8) = pPixelBlockSol7.GetVal(0, x, y)
    pFeatureBuffer.Value(9) = pPixelBlockSol8.GetVal(0, x, y)
    pFeatureBuffer.Value(10) = pPixelBlockSol9.GetVal(0, x, y)
    pFeatureBuffer.Value(11) = pPixelBlockSol10.GetVal(0, x, y)
    pFeatureBuffer.Value(12) = pPixelBlockSol11.GetVal(0, x, y)
    pFeatureBuffer.Value(13) = pPixelBlockSol12.GetVal(0, x, y)
    pFeatureBuffer.Value(14) = pPixelBlockDur1.GetVal(0, x, y)
    pFeatureBuffer.Value(15) = pPixelBlockDur2.GetVal(0, x, y)
    pFeatureBuffer.Value(16) = pPixelBlockDur3.GetVal(0, x, y)
    pFeatureBuffer.Value(17) = pPixelBlockDur4.GetVal(0, x, y)
    pFeatureBuffer.Value(18) = pPixelBlockDur5.GetVal(0, x, y)
    pFeatureBuffer.Value(19) = pPixelBlockDur6.GetVal(0, x, y)
    pFeatureBuffer.Value(20) = pPixelBlockDur7.GetVal(0, x, y)
    pFeatureBuffer.Value(21) = pPixelBlockDur8.GetVal(0, x, y)
    pFeatureBuffer.Value(22) = pPixelBlockDur9.GetVal(0, x, y)
    pFeatureBuffer.Value(23) = pPixelBlockDur10.GetVal(0, x, y)
    pFeatureBuffer.Value(24) = pPixelBlockDur11.GetVal(0, x, y)
    pFeatureBuffer.Value(25) = pPixelBlockDur12.GetVal(0, x, y)
```

```
  End If
```

```
  If pFeatureBuffer.Value(2) >= 0 Or _
    pFeatureBuffer.Value(3) >= 0 Or _
    pFeatureBuffer.Value(4) >= 0 Or _
    pFeatureBuffer.Value(5) >= 0 Or _
    pFeatureBuffer.Value(6) >= 0 Or _
    pFeatureBuffer.Value(7) >= 0 Or _
    pFeatureBuffer.Value(8) >= 0 Or _
    pFeatureBuffer.Value(9) >= 0 Or _
    pFeatureBuffer.Value(10) >= 0 Or _
    pFeatureBuffer.Value(11) >= 0 Or _
    pFeatureBuffer.Value(13) >= 0 Or _
    pFeatureBuffer.Value(13) >= 0 Or _
    pFeatureBuffer.Value(14) >= 0 Or _
    pFeatureBuffer.Value(15) >= 0 Or _
    pFeatureBuffer.Value(16) >= 0 Or _
    pFeatureBuffer.Value(17) >= 0 Or _
    pFeatureBuffer.Value(18) >= 0 Or _
    pFeatureBuffer.Value(19) >= 0 Or _
    pFeatureBuffer.Value(20) >= 0 Or _
    pFeatureBuffer.Value(21) >= 0 Or _
    pFeatureBuffer.Value(22) >= 0 Or _
    pFeatureBuffer.Value(23) >= 0 Or _
    pFeatureBuffer.Value(24) >= 0 Or _
    pFeatureBuffer.Value(25) >= 0 Then
```

```
pFCursor.InsertFeature pFeatureBuffer
flushCount = flushCount + 1
If CLng(flushCount / 20000) = 1 Then
    Debug.Print "flush " & Now; pbCount; x; y
    pFCursor.Flush
    flushCount = 0
End If
End If
currX = currX + increment

Next x
Debug.Print Now
currY = currY - increment
currX = startX
Next y

pbCount = pbCount + 1

Loop Until Not pRasterCursorDur1.Next
pFCursor.Flush
End Sub
```